

Дискреционное разграничение прав в Linux. Исследование влияния дополнительных атрибутов

Кузьмин

24 марта, 2025, Москва, Россия

Российский Университет Дружбы Народов

Цели и задачи

- SUID - разрешение на установку идентификатора пользователя. Это бит разрешения, который позволяет пользователю запускать исполняемый файл с правами владельца этого файла.
- SGID - разрешение на установку идентификатора группы. Принцип работы очень похож на SUID с отличием, что файл будет запускаться пользователем от имени группы, которая владеет файлом.

Цель лабораторной работы

Изучение механизмов изменения идентификаторов, применения SetUID и Sticky-битов. Получение практических навыков работы в консоли с дополнительными атрибутами. Рассмотрение работы механизма смены идентификатора процессов пользователей, а также влияние бита Sticky на запись и удаление файлов.

Выполнение лабораторной работы

Программа simpleid

Рис. 1: результат программы simpleid

Программа simpleid2

The screenshot shows a Windows terminal window with the following content:

```

C:\Users\user> gcc simpleid2.c
C:\Users\user> gcc simpleid2.c -o simpleid2
C:\Users\user> .\simpleid2
10
55

```

The program code is as follows:

```

#include <stdio.h>
#include <pthread.h>

int fib(int n) {
    if (n == 0) return 0;
    if (n == 1) return 1;
    return fib(n-1) + fib(n-2);
}

int main() {
    int n;
    scanf("%d", &n);
    printf("Fibonacci(%d) = %d\n", n, fib(n));
    return 0;
}

```

Рис. 2: результат программы simpleid2

Программа readfile

```

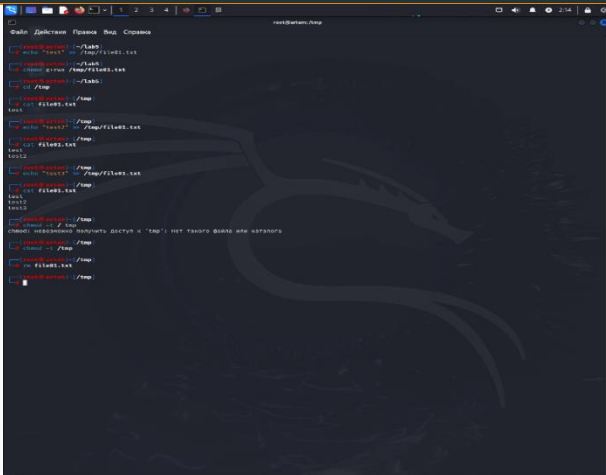
1  2  3  4  5  6  7  8  9  10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100
root@term-vuln:~# cat main.c
#include <stdio.h>
#include <string.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>

int main(int argc, char *argv[])
{
    unsigned char buffer[10];
    if(argc > 1) {
        int fd = open(argv[1], O_RDONLY);
        if(fd == -1) {
            perror("open");
            return 1;
        }
        ssize_t nread;
        while((nread = read(fd, buffer, sizeof(buffer))) > 0) {
            printf("%s\n", buffer);
        }
        close(fd);
    } else {
        // Read from stdin
        while(1) {
            ssize_t nread;
            while((nread = read(STDIN_FILENO, buffer, sizeof(buffer))) > 0) {
                printf("%s\n", buffer);
            }
        }
    }
}

```

Рис. 3: результат программы readfile

Исследование Sticky-бита



```
root@ubuntu: /tmp
[~] root@ubuntu: ~ /tmp
# echo "test1" > /tmp/file01.txt
[~] root@ubuntu: ~ /tmp
# cp /tmp /tmp/file02.txt
[~] root@ubuntu: ~ /tmp
# cd /tmp
[~] root@ubuntu: /tmp
# cat file01.txt
test1
[~] root@ubuntu: /tmp
# echo "test2" > /tmp/file02.txt
[~] root@ubuntu: /tmp
# cat file02.txt
test1
test2
[~] root@ubuntu: /tmp
# echo "test3" > /tmp/file03.txt
[~] root@ubuntu: /tmp
# cat file03.txt
test1
test2
test3
[~] root@ubuntu: /tmp
# ls -ld /tmp
drwxr-xr-x  2 root root 4096 ноя 14 22:14 /tmp
[~] root@ubuntu: /tmp
# rm /tmp
rm: cannot remove '/tmp': нет такого файла или каталога
[~] root@ubuntu: /tmp
# rm /tmp
rm: cannot remove '/tmp': нет такого файла или каталога
[~] root@ubuntu: /tmp
# ls -ld /tmp
drwxr-xr-x  2 root root 4096 ноя 14 22:14 /tmp
```

Рис. 4: исследование Sticky-бита

Выводы

Результаты выполнения лабораторной работы

Изучили механизмы изменения идентификаторов, применения SetUID- и Sticky-битов. Получили практические навыки работы в консоли с дополнительными атрибутами. Также мы рассмотрели работу механизма смены идентификатора процессов пользователей и влияние бита Sticky на запись и удаление файлов.